

Laboratorium 1

Łuszczek Patryk
272707

5 maja 2025

1 Cel zadania

Celem zadania było stworzenie aplikacji typu PWA - Progressive Web App oraz wdrożenie jej w system chmurowy. Progressive Web App to aplikacja internetowa, która wykorzystuje możliwości nowoczesnych przeglądarek internetowych, aby zapewnić użytkownikom doświadczenie podobne do aplikacji natywnych. Aplikacje PWA są responsywne, szybkie i mogą działać offline, co czyni je idealnym rozwiązaniem dla użytkowników mobilnych.

2 Opis aplikacji

Aplikacja służy do proponowania przepisów kulinarnych pobieranych z API dostępnego pod adresem <https://www.themealdb.com>. Użytkownik po wejściu do aplikacji może przeglądać losowo pobierane przepisy oraz zapisywać je do ulubionych. Przepisy zapisane do ulubionych są następnie przechowywane w pamięci, co pozwala na ich późniejsze przeglądanie nawet w trybie offline.

3 Opis procesu

3.1 Tworzenie aplikacji

W celu umożliwienia działania aplikacji jako PWA został utworzony "service worker", który obsługuje cache'owanie zasobów aplikacji oraz zapewnienie możliwości działania offline. W czasie instalacji aplikacji SW zapisuje do pamięci podręcznej zdefiniowane pliki, w tym przypadku były to pliki `index.html`, `style.css`, `main.js`. Równie ważnym krokiem było utworzenie pliku manifestu, który definiuje podstawowe informacje aplikacji takie jak nazwa, ikona czy kolor motywu.

```

const cacheName = "piac-pwa-v1";
const baseUrl = "/pwa-js/"
const filesToCache = [
  baseUrl + "index.html",
  baseUrl + "manifest.json",
  baseUrl + "images/icons/icon-192x192.png",
  baseUrl + "styles.css",
  baseUrl + "js/main.js"
]

self.addEventListener("install", (event) => {
  event.waitUntil(
    caches.open(cacheName).then((cache) => {
      return cache.addAll(filesToCache).catch(err => {
        console.error("Caching failed:", err);
      });
    })
  );
});

self.addEventListener("fetch", (event) => {
  event.respondWith(
    caches.match(event.request).then((response) => {
      if (response) {
        return response;
      }
      return fetch(event.request).then((fetchResponse) => {
        if (event.request.method === "GET") {
          return caches.open(cacheName).then((cache) => {
            cache.put(event.request, fetchResponse.clone());
            return fetchResponse;
          });
        }
      });
    })
  );
});

```

Rysunek 1: Przykład pliku manifestu